mkvbatchmultiplex

Release 2.0.2

Dec 18, 2020

This project started out of the need to multiplex a few hundreds episodes of series that where encoded using AVI container and SRT for subtitles. In order to be maintained by a media server and present better information for audio and subtitle language.

CHAPTER 1

Installation

I'm working on the backbone to maintain the project on GitHub once is up work on easy installations for the supported operating system will be made. If you are a python user I don't see any problems you using the application right away.

1.1 Python package

Python users can install the program using pip python dependencies will automatically install if not installed.

```
pip install mkvbatchmultiplex
```

1.2 Dependencies

The program uses:

Python packages are installed if pip is used:

natsort - natural sorting in Python https://natsort.readthedocs.io/en/master/

pymediainfo - Python MediaInfo wrapper, you can find it at https://pypi.org/project/pymediainfo/

PySide2 - GUI interface library used, https://wiki.qt.io/Qt_for_Python

MKVToolNix - The target tool from witch we get the command, https://mkvtoolnix.download/

MediaInfo library, witch you can find at https://mediaarea.net/en/MediaInfo. The library has to be available in the PATH environment variable.

For macOS and Windows the version of pymediainfo comes with a MediaInfo Linux does not.

1.3 Binary installations

Binary packages can be found:

> https://github.com/akai10tsuki/mkvbatchmultiplex/releases
>
> Linux and macOS binaries need more testing.

All binaries are for 64 bit architecture.

Windows:

- MKVBatchMultiplex-iss-AMD64-X.X.X.exe - Regular installation file
- MKVBatchMultiplex-X.X.X-portable.exe - Portable executable
- MKVBatchMultiplex-portable-dir.zip - Portable version in directory format may load faster that the standalone version

macOS:

- MKVBatchMultiplex-X.X.X-cmp.dmg - Disk image just mount and drag application to Applications folder

Linux:

- MKVBatchMultiplex-x86_64-X.X.X.AppImage - AppImage of the application is a portable version. Make it an executable file in order to run it. - install MediaInfo library

1.4 Known Issues

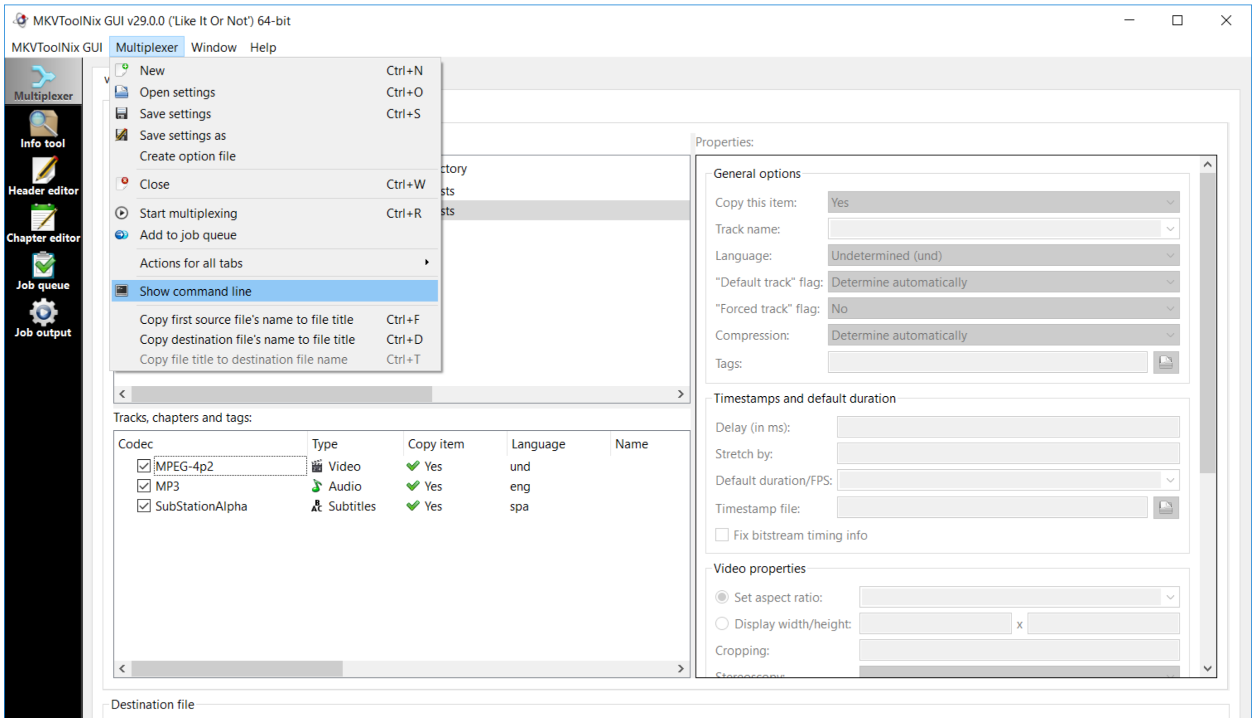
- **On macOS 10.14 Mojave:**
  - dark theme in MKVToolNix prior to 30.0.0 doesn't work.

CHAPTER 2

Getting Started

Using **MKVToolNix** do any needed operations on the first media file in the source directory. When done copy the resulting command line to the clipboard:

Multiplexer->Show command line



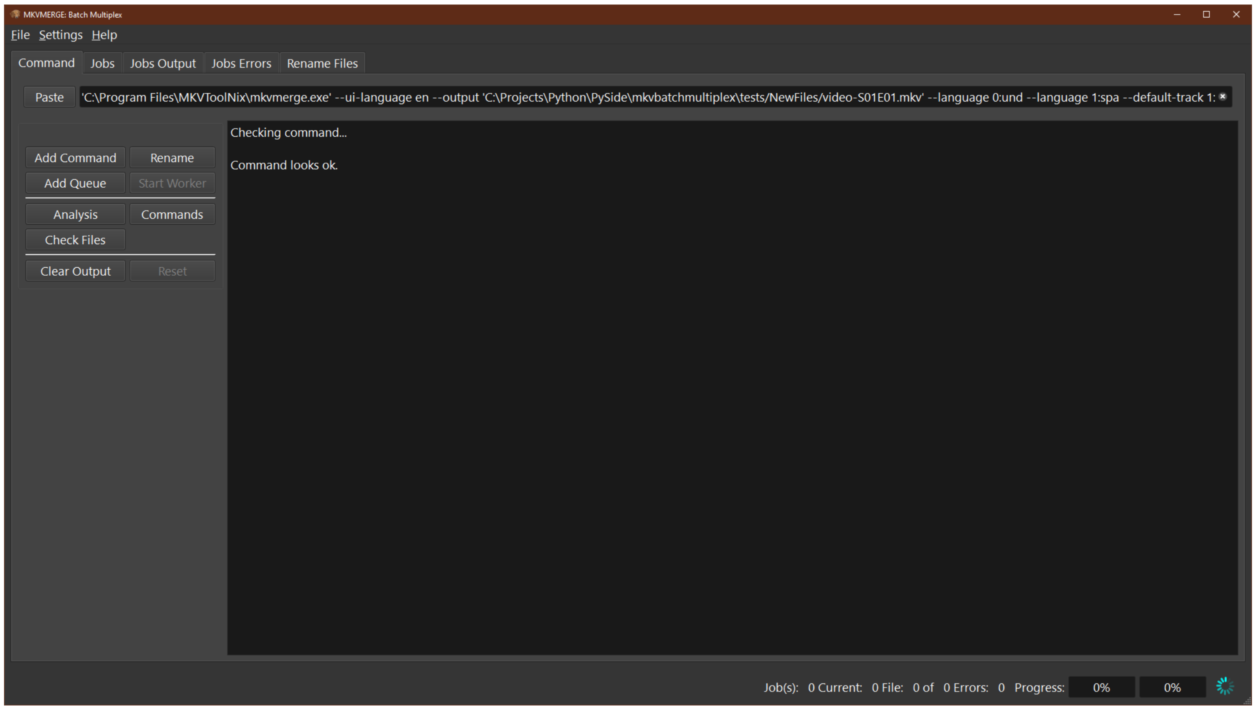
Copy this item: Yes
Track name:
Language: Undetermined (und)
"Default track" flag: Determine automatically
"Forced track" flag: No
Compression: Determine automatically
Tags:
Delay (in ms):
Stretch by:
Default duration/FPS:
Timestamp file:
Fix bitstream timing info
Set aspect ratio:
Display width/height: x
Cropping:

Codec	Type	Copy item	Language	Name
☑ MPEG-4p2	Video	✓ Yes	und	
☑ MP3	Audio	✓ Yes	eng	
☑ SubStationAlpha	Subtitles	✓ Yes	spa	

Fig. 1: Copy to clipboard

Open mkvbatchmultiplex and paste command using the **<Paste>** button:

Now there are two options **<Add Command>** button will add the job to the Jobs Table with a 'Waiting' status. The



```
'C:\Program Files\MKVToolNix\mkvmerge.exe' --ui-language en --output 'C:\Projects\Python\PySide\mkvbatchmultiplex\tests/NewFiles/video-S01E01.mkv' --language 0:und --language 1:spa --default-track 1:
```

```
Checking command...

Command looks ok.
```

Fig. 2: Copy from clipboard

<Add Queue> button will add the command to the job Queue. When finished adding jobs to the Queue push **<Start Worker>** to start the Queue worker and run the jobs. Any job added with the ‘Waiting’ status have to be added to the Queue in the Jobs tab in order to run the job. If the Queue worker is running jobs added to the Queue will be processed in the order entered.

CHAPTER 3

Using mkvbatchmultiplex

The application has a very simple interface:

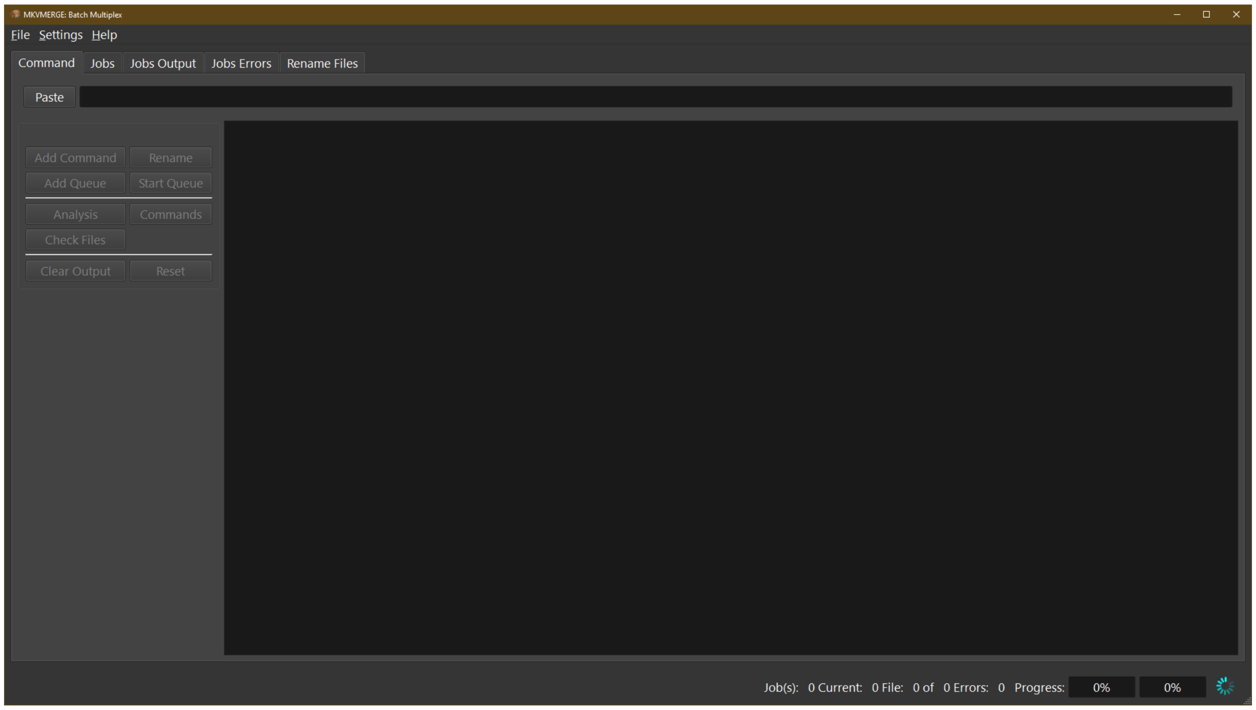

MKVMERGE: Batch Multiplex
File Settings Help
Command Jobs Jobs Output Jobs Errors Rename Files
Paste
Add Command Rename
Add Queue Start Queue
Analysis Commands
Check Files
Clear Output Reset
Job(s): 0 Current: 0 File: 0 of 0 Errors: 0 Progress: 0% 0%

Fig. 1: mkvbatchmultiplex

3.1 Requirements

In order for mkvbatchmultiplex to work as intended there are certain conditions in the structure of the files in the source directories that need to be met. This has to be this way because we are working with group of files not one file. Also is a batch process there should not be any user input requested when working of a job. For the job to succeed the corresponding file have to be equal in the internal structure tracks have to be in the same order and same type.

Structure for directories:

- The media files in the directory have to be of the same type. The program will read the directory taking information form the template so it will apply instruction to the same type of file.
- The files have to be consistent that's to say the same number of tracks, same type and order. Files found different from template will not be processed.
- If more than one file are involved in the operation there have to be one of each for the resulting tracks. For example:

  The instruction if for a video file and a subtitle file to be multiplexed and there are 13 video files then there have to be 13 subtitle files.

  One source directory:

/source1	/Destination
video - S01E01.avi	video - S01E01.mkv
video - S01E01.srt	
video - S01E02.avi	video - S01E02.mkv
video - S01E02.srt	
…	…
…	…
video - S01E13.avi	video - S01E13.mkv
video - S01E13.srt	

- For attachments the same files will be use for every operation. If the attachments are per file the files have to be on a directory with one directory per file. If a file does not have attachments the directory still has to exists but empty.
  - If more than one directory is used to get the attachments then the same attachments will be use for all the files. The directory per file will not be check or used.
- Files of the same type in the command have to be in different directories. To process the files names are not used to pair the sources in alphabetical position is:

  Source1 AVI container source2 lang1.srt source3 lan2.srt:

  Two source directory:

/directory1	/directory2	/destination
video - S01E01.avi	video - S01E01.srt	video - S01E01.mkv
video - S01E02.avi	video - S01E02.srt	video - S01E02.mkv
…	…	…
video - S01E13.avi	video - S01E13.srt	video - S01E13.mkv

  This the best option for the working directories.

3.2 Functionality

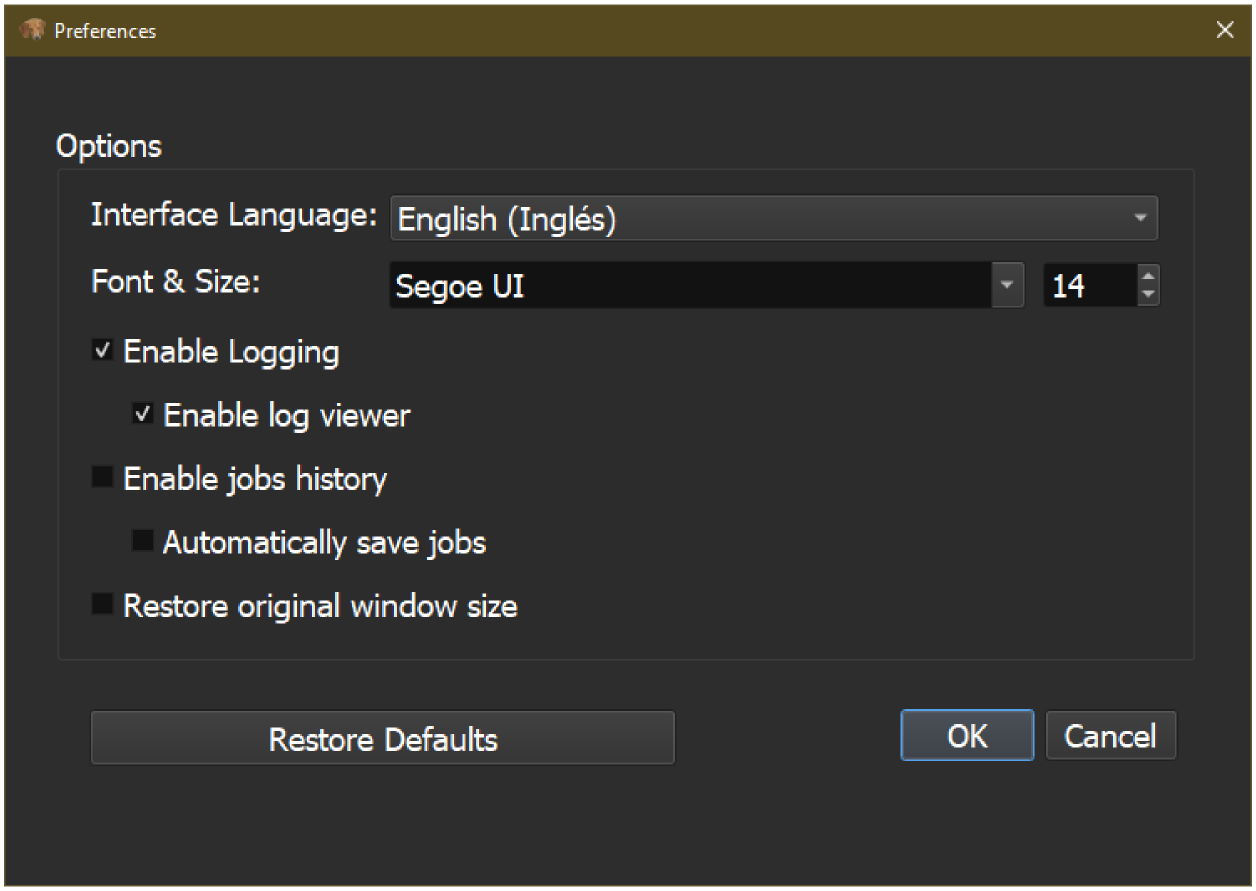
3.2.1 Source File Naming

The source and destination directories are taken from the template. The resulting name in the directory is taken from the first media file selected for the multiplexing. The name for the subsequent files is not used what is use is the order read. The program will pair the files by order found in the directory:

/directory1	/directory2	/destination
video - S01E01.avi	sub01.srt	video - S01E01.mkv
video - S01E02.avi	sub02.srt	video - S01E02.mkv
…	…	…
video - S01E13.avi	sub03.srt	video - S01E13.mkv

If the destination directory already have a file with the destination name a prefix **new-** will be used. The job operation is not destructive no file will be overwritten.

3.2.2 Settings



Interface Language: English (Inglés)

Font & Size: Segoe UI 14

✓ Enable Logging

✓ Enable log viewer

Enable jobs history

Automatically save jobs

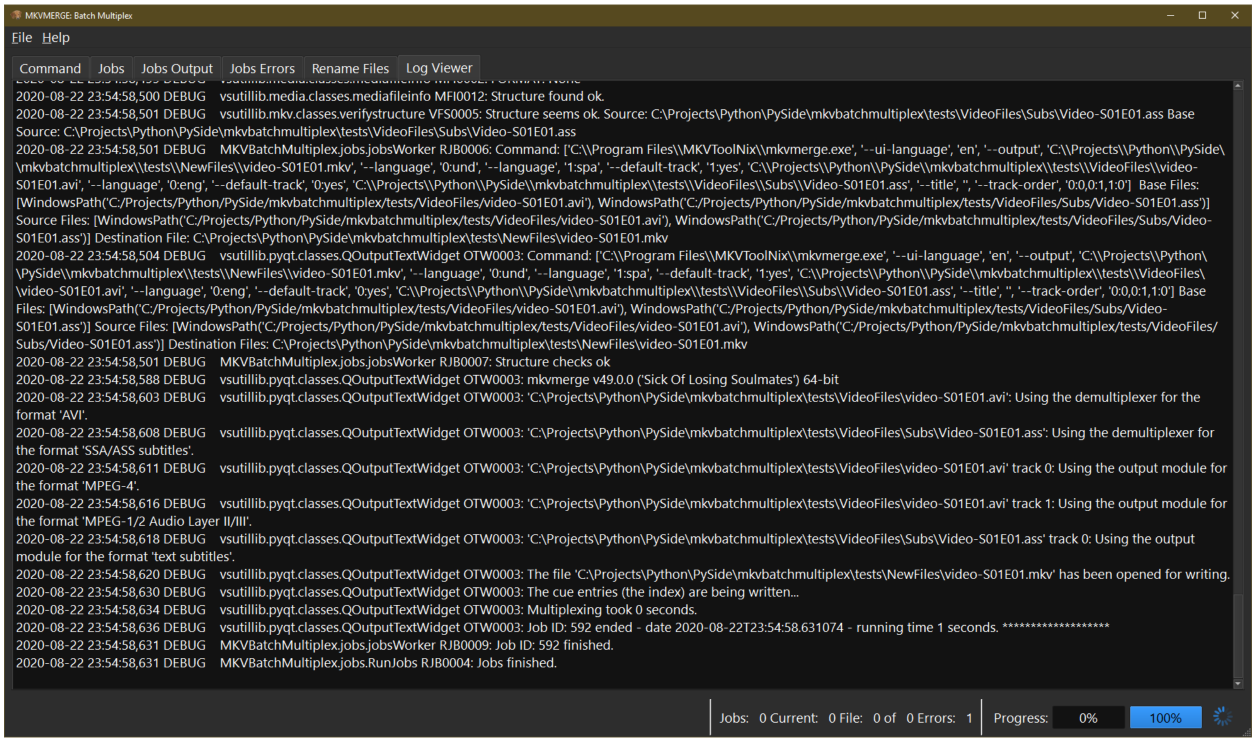
Restore original window size

Fig. 2: Settings.

The available settings:

- Interface Language:
  - English
  - Español
- Enable logging. If enable logging is set the log will be saved on:
  - ~/.mkvbatchmultiplex/mkvBatchMultiplex.log

  with a rotation of 10 log files.
  - Enable log viewer. If enabled there will be a tab where the log can be seen as the program runs.
- Font & Size. Change the font use by the interface.
- Interface Language. Change the language use by the interface. English and Spanish are available.
- Restore Defaults. Restore default settings.


MKVMERGE: Batch Multiplex
File Help
Command Jobs Jobs Output Jobs Errors Rename Files Log Viewer

```
2020-08-22 23:54:58,500 DEBUG    vsutillib.media.classes.mediafileinfo MFI0012: Structure found ok.
2020-08-22 23:54:58,501 DEBUG    vsutillib.mkv.classes.verifystructure VFS0005: Structure seems ok. Source: C:\Projects\Python\PySide\mkvbatchmultiplex\tests\VideoFiles\Subs\Video-S01E01.ass Base Source: C:\Projects\Python\PySide\mkvbatchmultiplex\tests\VideoFiles\Subs\Video-S01E01.ass
2020-08-22 23:54:58,501 DEBUG    MKVBatchMultiplex.jobs.jobsWorker RJB0006: Command: ['C:\\Program Files\\MKVToolNix\\mkvmerge.exe', '--ui-language', 'en', '--output', 'C:\\Projects\\Python\\PySide\\mkvbatchmultiplex\\tests\\NewFiles\\video-S01E01.mkv', '--language', '0:und', '--language', '1:spa', '--default-track', '1:yes', 'C:\\Projects\\Python\\PySide\\mkvbatchmultiplex\\tests\\VideoFiles\\video-S01E01.avi', '--language', '0:eng', '--default-track', '0:yes', 'C:\\Projects\\Python\\PySide\\mkvbatchmultiplex\\tests\\VideoFiles\\Subs\\Video-S01E01.ass', '--title', '', '--track-order', '0:0,0:1,1:0']  Base Files: [WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/video-S01E01.avi'), WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/Subs/Video-S01E01.ass')] Source Files: [WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/video-S01E01.avi'), WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/Subs/Video-S01E01.ass')] Destination File: C:\Projects\Python\PySide\mkvbatchmultiplex\tests\NewFiles\video-S01E01.mkv
2020-08-22 23:54:58,504 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: Command: ['C:\\Program Files\\MKVToolNix\\mkvmerge.exe', '--ui-language', 'en', '--output', 'C:\\Projects\\Python\\PySide\\mkvbatchmultiplex\\tests\\NewFiles\\video-S01E01.mkv', '--language', '0:und', '--language', '1:spa', '--default-track', '1:yes', 'C:\\Projects\\Python\\PySide\\mkvbatchmultiplex\\tests\\VideoFiles\\video-S01E01.avi', '--language', '0:eng', '--default-track', '0:yes', 'C:\\Projects\\Python\\PySide\\mkvbatchmultiplex\\tests\\VideoFiles\\Subs\\Video-S01E01.ass', '--title', '', '--track-order', '0:0,0:1,1:0'] Base Files: [WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/video-S01E01.avi'), WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/Subs/Video-S01E01.ass')] Source Files: [WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/video-S01E01.avi'), WindowsPath('C:/Projects/Python/PySide/mkvbatchmultiplex/tests/VideoFiles/Subs/Video-S01E01.ass')] Destination Files: C:\Projects\Python\PySide\mkvbatchmultiplex\tests\NewFiles\video-S01E01.mkv
2020-08-22 23:54:58,501 DEBUG    MKVBatchMultiplex.jobs.jobsWorker RJB0007: Structure checks ok
2020-08-22 23:54:58,588 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: mkvmerge v49.0.0 ('Sick Of Losing Soulmates') 64-bit
2020-08-22 23:54:58,603 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: 'C:\Projects\Python\PySide\mkvbatchmultiplex\tests\VideoFiles\video-S01E01.avi': Using the demultiplexer for the format 'AVI'.
2020-08-22 23:54:58,608 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: 'C:\Projects\Python\PySide\mkvbatchmultiplex\tests\VideoFiles\Subs\Video-S01E01.ass': Using the demultiplexer for the format 'SSA/ASS subtitles'.
2020-08-22 23:54:58,611 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: 'C:\Projects\Python\PySide\mkvbatchmultiplex\tests\VideoFiles\video-S01E01.avi' track 0: Using the output module for the format 'MPEG-4'.
2020-08-22 23:54:58,616 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: 'C:\Projects\Python\PySide\mkvbatchmultiplex\tests\VideoFiles\video-S01E01.avi' track 1: Using the output module for the format 'MPEG-1/2 Audio Layer II/III'.
2020-08-22 23:54:58,618 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: 'C:\Projects\Python\PySide\mkvbatchmultiplex\tests\VideoFiles\Subs\Video-S01E01.ass' track 0: Using the output module for the format 'text subtitles'.
2020-08-22 23:54:58,620 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: The file 'C:\Projects\Python\PySide\mkvbatchmultiplex\tests\NewFiles\video-S01E01.mkv' has been opened for writing.
2020-08-22 23:54:58,630 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: The cue entries (the index) are being written...
2020-08-22 23:54:58,634 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: Multiplexing took 0 seconds.
2020-08-22 23:54:58,636 DEBUG    vsutillib.pyqt.classes.QOutputTextWidget OTW0003: Job ID: 592 ended - date 2020-08-22T23:54:58.631074 - running time 1 seconds. ******************
2020-08-22 23:54:58,631 DEBUG    MKVBatchMultiplex.jobs.jobsWorker RJB0009: Job ID: 592 finished.
2020-08-22 23:54:58,631 DEBUG    MKVBatchMultiplex.jobs.RunJobs RJB0004: Jobs finished.
```

Fig. 3: Log viewer.

3.2.3 Tabs

- **Command**

  This is the main tab where the command is registered

  Command Tab buttons:

  - Paste

    Paste command line from system Clipboard is expected to be from a copy to clipboard from MKVToolnix.

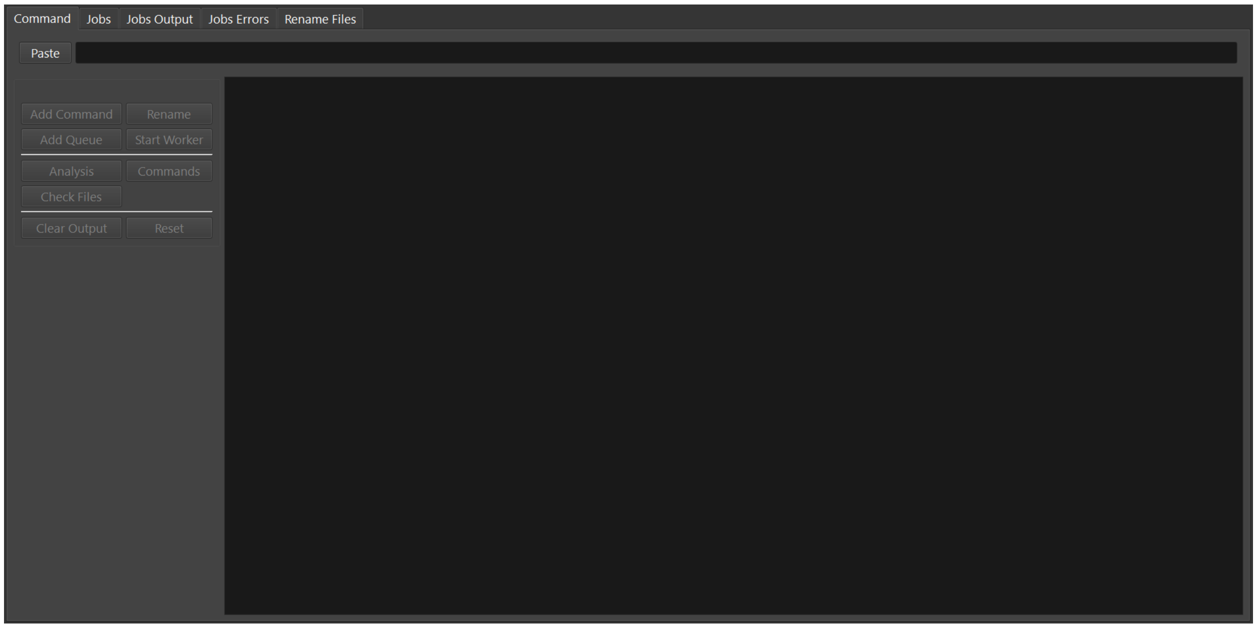


Fig. 4: Command tab.

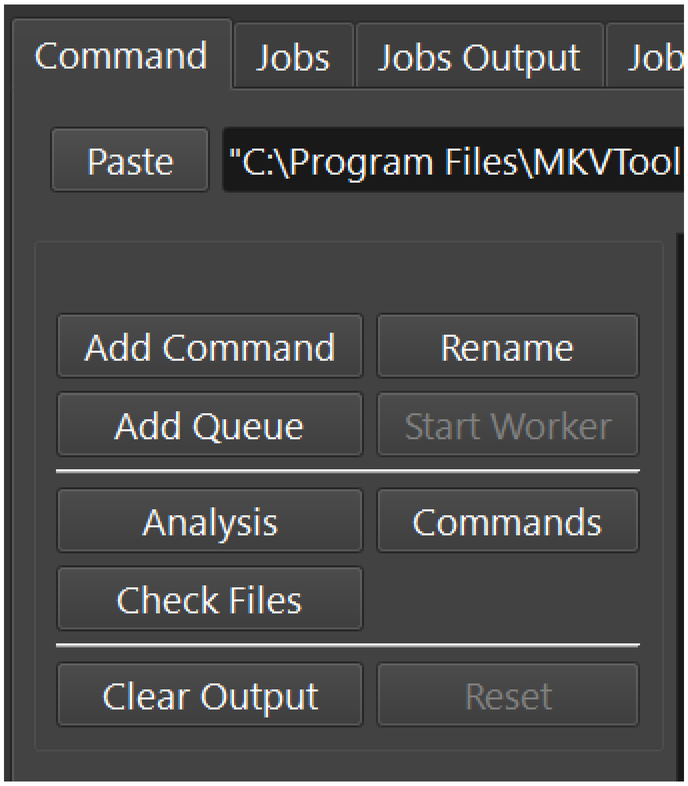


Fig. 5: Command.

  - Add Command

    Add the command to the jobs table with a **Waiting** status

  - Rename

    Go to the **Rename** tab

  - Add Queue

    Add command to the jobs table and the work Queue. If the job worker is running the job will be processed

  - Start Worker

    Start working on the jobs in the queue. While the jobs are being process more can be added to the queue.

  The following buttons are for troubleshooting:

  - Analysis

    Shows some information the command line parsing this can help solving problems as to why the command is not been accepted

  - Commands

    Show commands to be executed as lists to be submitted to a subprocess

  - Check Files

    Verify the files to be worked on and signals any command that don't pass consistency test. Flagged commands will not be processed.

  - Clear Output

    Clear the output of the command window

  - Reset

    Clear contents of all output windows

  When a command is entered it will be check for it to see if it is in the format the application can handle. A message **Command looks ok.** does not necessarily means it will work. There are other checks that are done when the job is executed. For example the number of input files does not match this will fail the entire job. This is done like this because the operation is time consuming and the GUI will seems to be frozen. When some files don't meet criteria those files won't be processed but the ones that match will. When the job fails use the buttons for troubleshooting to see if the problem is revealed. If still cannot find the problem post an issue to get help.

  Before adding the job to the queue you have the option to push **<Rename>** to go to the **Rename** tab and rename the output files.

- **Jobs**

  Displays a table with the jobs added for current session.

  Jobs **Status** column. The jobs can be manipulated using this column:

  - **Waiting**: the job has to be added to the queue for processing. While the job is waiting double click it can be changed to **Skip**, meaning don't process the job. Or using the push button **<Queue Waiting Jobs>** it will add the job to the job queue.
  - **Running**: this is the current running job.
  - **Done**: job already processed.
  - **Skip**: job will not be processed. While in Skip status the job can be changed back to previous status of **Waiting** or **Queue** if the worker is active it will be processed automatically.
  - **Aborted**: the job was stopped while it was running.

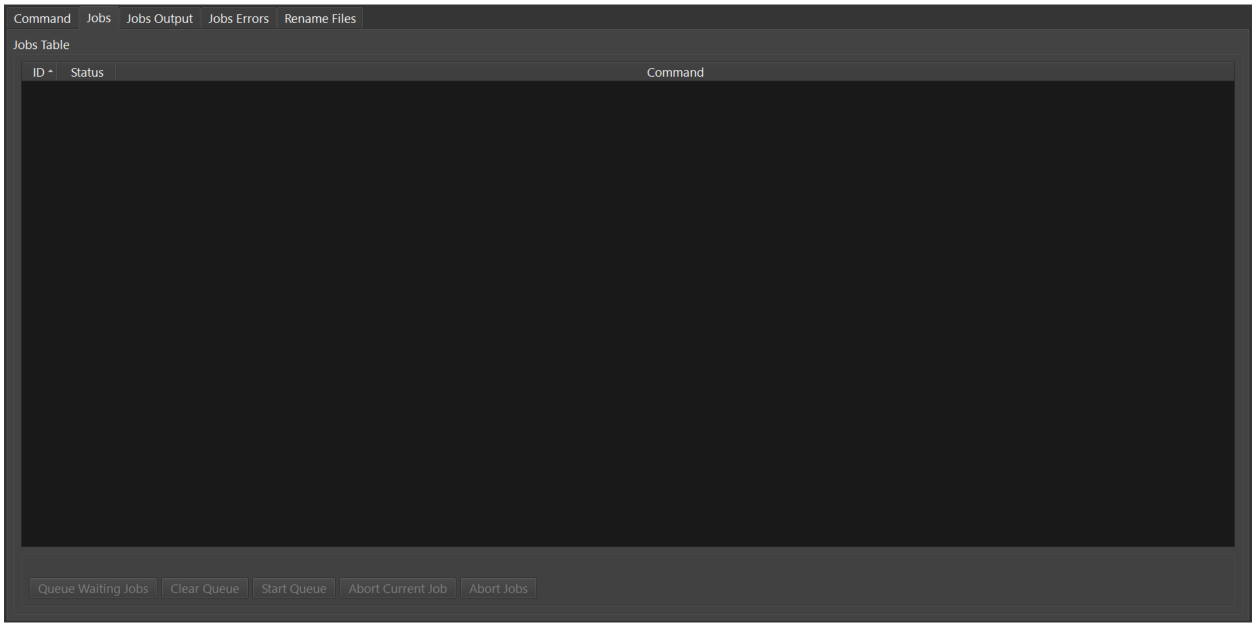


Fig. 6: Jobs.

- **Error**: and error ocurred while running. A full destination disk can cause this.

Jobs tab buttons:

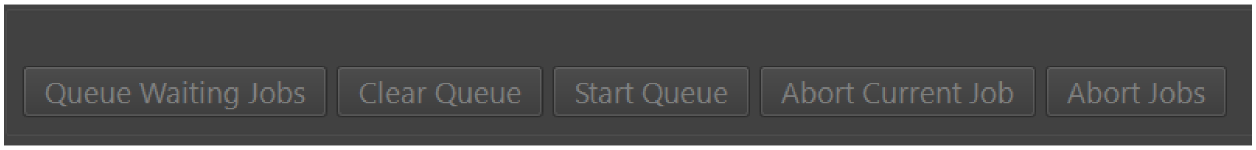


Fig. 7: Jobs buttons.

- Queue Waiting Jobs

  Put all **Waiting** jobs on the Queue for processing

- Clear Queue

  Set all jobs in the Queue to a Waiting status. The Worker will not process them.

- Star Worker

  Start the Worker and process all jobs on the Queue.

- Abort Current Jobs

  Abort the current job immediately only the current file been processed will be deleted. All finished files will not.

- Abort Jobs

  Abort running jobs and all jobs on the Queue.

- **Jobs Output**

  Displays output messages generated by the running jobs.

```
mkvmerge v30.0.0 ('Interstellar') 64-bit
'\video'S01E05.avi': Using the demultiplexer for the format 'AVI'.
'\Video'S01E05.ass': Using the demultiplexer for the format 'SSA/ASS subtitles'.
'\video'S01E05.avi' track 0: Using the output module for the format 'MPEG-4'.
'\video'S01E05.avi' track 1: Using the output module for the format 'MP3'.
'\Video'S01E05.ass' track 0: Using the output module for the format 'text␣
↪subtitles'.
The file 'C:\tests\NewFiles\new-video'S01E05 (15).mkv' has been opened for␣
↪writing.
Progress: 100%

The cue entries (the index) are being written...
Multiplexing took 0 seconds.
```

- **Jobs Errors**

  Display any errors found generally this means any files with inconsistent with original files track order or type don't match.

```
Error Job ID: 286 ---------------------

Destination File: C:\tests\NewFiles\video - S01E04.mkv

Error: In structure

Source:
File Nme: C:\tests\VideoFiles\video - S01E04.avi
File Format: -AVI-

Track: 1
Order: 0 - Video
Codec: None
Language: None
Format: MPEG-4 Visual

Base Source:
File Nme: C:\tests\VideoFiles\video-S01E01.avi
File Format: -AVI-

Track: 1
Order: 0 - Video
Codec: None
Language: None
Format: MPEG-4 Visual
Track: 2
Order: 1 - Audio
Codec: None
Language: None
Format: MPEG Audio

Number of tracks mismatched video - S01E04.avi: 1 - video-S01E01.avi: 2

Error Job ID: 286 ---------------------
```

- **Rename Files**

  The Rename module uses python regular expressions witch is considered to be for advance users. In order to help and make it as easy as possible regular expressions that cover a great number of the cases presented to me for downloaded series provided. Also using '*' (asterisk) by itself as the regular expression a new name can be easily assigned.

For the majority of media servers a good naming scheme for the episodes of a series is:

```
Series Name - S01E01.mkv
```

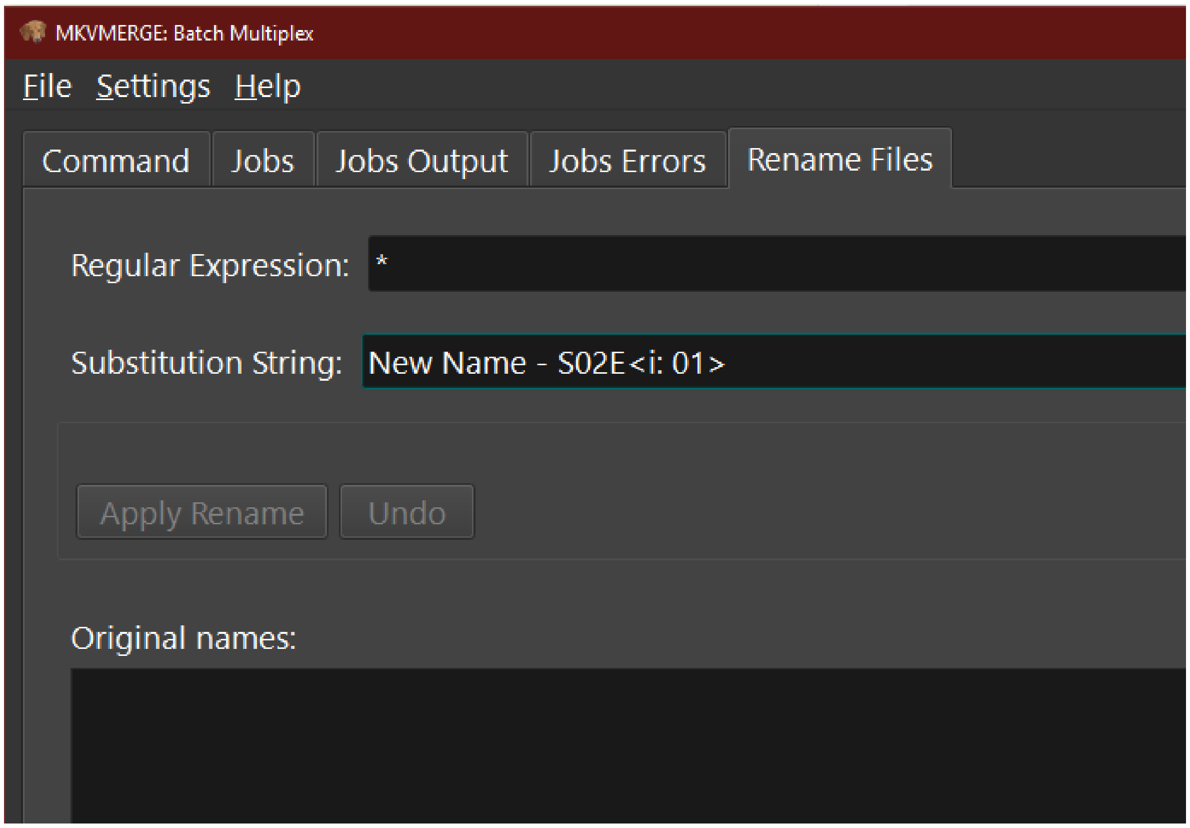
The part S01E01 represents season 1 episode 1. If the files been process don't follow this scheme the system provides the ability to rename the output files. Using regular expressions you can rename the output file names. Also if regular expressions are to difficult a new name with and index in the form:

```
Series Name - S01E<i: NN>
```

<i: NN> where NN is a number can be 0 padded. It will be substituted by a number starting with the value NN.

As shown in the figure to rename using and index set the regex to * (asterisk, and invalid regex to use). Then enter the new name with the increment mark in the desired position. Once satisfied push button Apply Rename

Example:


MKVMERGE: Batch Multiplex
File Settings Help
Command Jobs Jobs Output Jobs Errors Rename Files
Apply Rename Undo
Original names:

Regular Expression: *

Substitution String: New Name - S02E<i: 01>

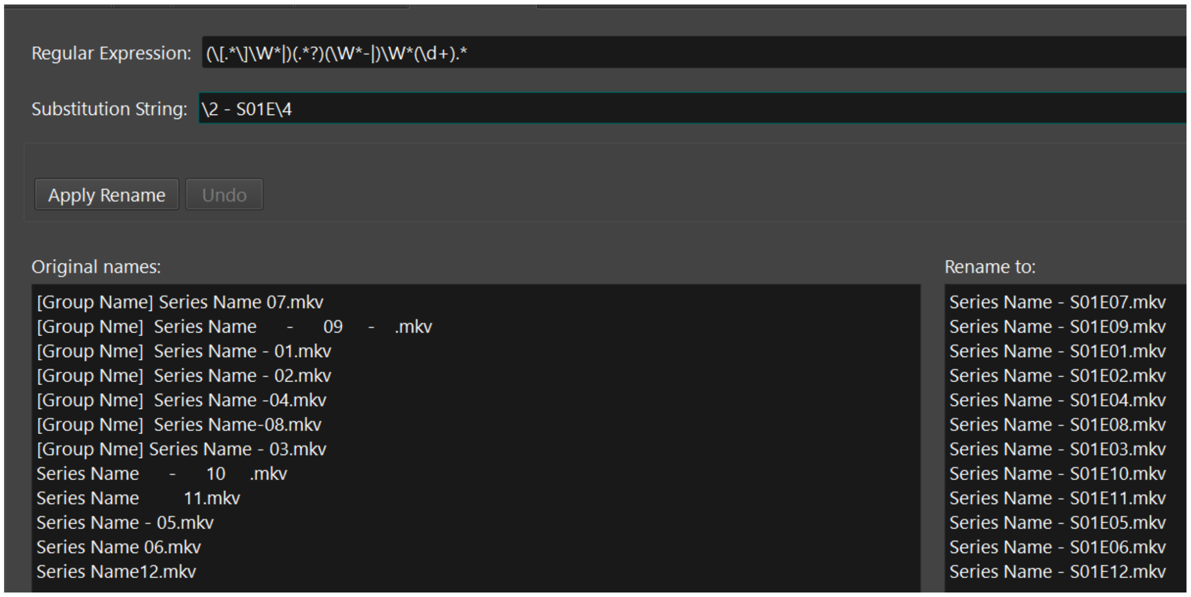
Fig. 8: Rename with incremental index.

The operation can be undone with the **Undo** pushbutton before starting the execution of the batch operation.

If it proves to difficult use the '*' with Series Name S01E<i: 01> for renaming. Also I can help creating the regex by sending me the list of the original names and a template of the desired new name.

3.3 Know Issues

Work on documentation.



```
(\[.*\]\W*|)(.*?)(\W*-|)\W*(\d+).*
```

```
\2 - S01E\4
```

```
[Group Name] Series Name 07.mkv
[Group Nme]  Series Name    -    09    -    .mkv
[Group Nme]  Series Name - 01.mkv
[Group Nme]  Series Name - 02.mkv
[Group Nme]  Series Name -04.mkv
[Group Nme]  Series Name-08.mkv
[Group Nme] Series Name - 03.mkv
Series Name     -     10    .mkv
Series Name        11.mkv
Series Name - 05.mkv
Series Name 06.mkv
Series Name12.mkv
```

```
Series Name - S01E07.mkv
Series Name - S01E09.mkv
Series Name - S01E01.mkv
Series Name - S01E02.mkv
Series Name - S01E04.mkv
Series Name - S01E08.mkv
Series Name - S01E03.mkv
Series Name - S01E10.mkv
Series Name - S01E11.mkv
Series Name - S01E05.mkv
Series Name - S01E06.mkv
Series Name - S01E12.mkv
```

Fig. 9: Rename with regex.

Here is an example where the name contains the series name and the episode number only.
The regular expression is::

```
(\\[.*\\]\\W*|)(.*?)(\\W*-|)\\W*(\\d+).*
```

The substitution string is::

```
\\2 - S01E\\4
```

For regular expressions the order is not important the episode number is taken from the name. Also missing episodes won't affect the rename of the files. What the regular expression is doing is creating 4 groups::

```
1. (\\[.*\\]\\W*|) - this will match the group name if Analysis
2. (.*?) - this will match the series name
3. (\\W*-|) - this will match a '-' hyphen witch normally
   separates the name from the episode number
4. (\\d+) - this will match the episode number
```

In the substitution string \N represent the group number \2 for series name \4 for episode number. The other characters are literals.

CHAPTER 4

Changelog

(Unreleased)

4.1 Added

4.2 Changed

4.3 Fixed

4.3.1 2.0.0 - 2000-8-23

4.4 Changed

- locale updates
- Check Files displays files read from the source directory. Also the contents of the destination directory for debug purposes.

4.5 Fixed

- python wheel distribution not working
- system tray icon not showing on macOS

4.5.1 2.0.0b1 - 2020-8-8

4.6 Added

- show progress bar on Windows taskbar icon
- view log on optional tab

4.7 Changed

- configuration now is a dialog for better compatibility with macOS
- use natural sort when reading directories

4.8 Fixed

- Fix BUG #1 force escape quotes for mkvmerge executable in Windows
- Fix BUG #3 title of first episode propagating to all episodes
- dummy progress bar icon function on macOS was not working
- removing configuration elements not always working
- spanish locale fixes

4.8.1 2.0.0a1 - 2019-12-5

- First release version 2.0
- Re-write of MKVBatchMultiplex
- Use a dark theme on Windows 10
- Add rename for output files
- Jobs table with jobs management
- Add Spanish Interface

CHAPTER 5

Contributing to MKVBatchMultiplex

We love your input! We want to make contributing to this project as easy and transparent as possible, whether it's:

- Reporting a bug
- Discussing the current state of the code
- Submitting a fix
- Proposing new features
- Becoming a maintainer

5.1 The Project is Develop with Github

Github is used to host code, to track issues and feature requests, as well as accept pull requests.

5.2 Github Flow is used, So All Code Changes Happen Through Pull Requests

Pull requests are the best way to propose changes to the codebase (Github Flow is used). Pull requests are welcome:

1. Fork the repo and create your branch from *master*.
2. If you've added code that should be tested, add tests.
3. If you've changed APIs, update the documentation.
4. Ensure the test suite passes.
5. Submit test folder listing with files and files description to replicate tests.
6. Make sure your code lints (pylint).
7. Code is formatted using black.

5.3 Any contributions you make will be understood under the MIT Software License

In short, when you submit code changes, your submissions are understood to be under the same MIT License that covers the project. Feel free to contact the maintainers if that's a concern.

5.4 Report bugs using Github's issues We use GitHub issues to track public bugs.

Report a bug by **opening a new issue**; it's that easy!

5.5 Write bug reports with detail, background, and sample code

Good Bug Reports tend to have:

- A quick summary and/or background
- Steps to reproduce
  - Be specific!
  - Program version.
  - Copy of command line.
  - List of files in the directories used. This is better achieved using the **<Check Files>** button and including the log file found in HOMEDIRECTORY/.MKVBatchMultiplex/MKVBatchMultiplex.log.
  - Structure of the files, list of tracks type and order in media files.
  - If tracks are raw please specify the type if it cannot be deduced by file extension.
- Include a log file.
- What you expected would happen
- What actually happens
- Notes (possibly including why you think this might be happening, or stuff you tried that didn't work). Also the activate and attach log file.

5.6 Use a Consistent Coding Style

The code is formatted using black and for linting pylint is used.

5.7 License

By contributing, you agree that your contributions will be licensed under its MIT License.

5.8 References

This document was adapted from the open-source contribution guidelines for Facebook's Draft

CHAPTER 6

Contributor Covenant Code of Conduct

6.1 Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

We pledge to act and interact in ways that contribute to an open, welcoming, diverse, inclusive, and healthy community.

6.2 Our Standards

Examples of behavior that contributes to a positive environment for our community include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback
- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

Examples of unacceptable behavior include:

- The use of sexualized language or imagery, and sexual attention or advances of any kind
- Trolling, insulting or derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or email address, without their explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

6.3 Enforcement Responsibilities

Community leaders are responsible for clarifying and enforcing our standards of acceptable behavior and will take appropriate and fair corrective action in response to any behavior that they deem inappropriate, threatening, offensive, or harmful.

Community leaders have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, and will communicate reasons for moderation decisions when appropriate.

6.4 Scope

This Code of Conduct applies within all community spaces, and also applies when an individual is officially representing the community in public spaces. Examples of representing our community include using an official e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event.

6.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported to the community leaders responsible for enforcement at akai10tsuki@gmail.com. All complaints will be reviewed and investigated promptly and fairly.

All community leaders are obligated to respect the privacy and security of the reporter of any incident.

6.6 Enforcement Guidelines

Community leaders will follow these Community Impact Guidelines in determining the consequences for any action they deem in violation of this Code of Conduct:

6.6.1 1. Correction

Community Impact: Use of inappropriate language or other behavior deemed unprofessional or unwelcome in the community.

Consequence: A private, written warning from community leaders, providing clarity around the nature of the violation and an explanation of why the behavior was inappropriate. A public apology may be requested.

6.6.2 2. Warning

Community Impact: A violation through a single incident or series of actions.

Consequence: A warning with consequences for continued behavior. No interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, for a specified period of time. This includes avoiding interactions in community spaces as well as external channels like social media. Violating these terms may lead to a temporary or permanent ban.

6.6.3 3. Temporary Ban

Community Impact: A serious violation of community standards, including sustained inappropriate behavior.

Consequence: A temporary ban from any sort of interaction or public communication with the community for a specified period of time. No public or private interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, is allowed during this period. Violating these terms may lead to a permanent ban.

6.6.4 4. Permanent Ban

Community Impact: Demonstrating a pattern of violation of community standards, including sustained inappropriate behavior, harassment of an individual, or aggression toward or disparagement of classes of individuals.

Consequence: A permanent ban from any sort of public interaction within the community.

6.7 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 2.0, available at https://www.contributor-covenant.org/version/2/0/code_of_conduct.html.

Community Impact Guidelines were inspired by Mozilla's code of conduct enforcement ladder.

For answers to common questions about this code of conduct, see the FAQ at https://www.contributor-covenant.org/faq. Translations are available at https://www.contributor-covenant.org/translations.

CHAPTER 7

License

CHAPTER 8

Acknowledgements

- Martin Fitzpatrick
  - Tutorial for the multithreading code